Fundamentos de Algoritmos

Grados en Ingeniería Informática

Convocatoria extraordinaria, 5 de julio de 2021. Grupos B y D

1. (3 puntos) Una secuencia de enteros se denomina *escalera* cuando cada entero, a excepción del primero, se obtiene sumando o restando 1 al anterior. Por otro lado, la *integral* de una secuencia de enteros es la suma de los elementos que la forman.

Desarrolla un algoritmo iterativo eficiente que, dado un vector no vacío de enteros no negativos determine la *integral* de la *escalera* con mayor *integral* de todas las contenidas en el vector. Aparte de implementar el algoritmo, debes indicar su precondición, su postcondición, el invariante del bucle y la función de cota que permiten verificar la corrección del algoritmo, y determinar razonadamente su complejidad.

La implementación deberá ir acompañada de un programa de prueba, que lea desde la entrada estandar casos de prueba, los ejecute, e imprima por la salida estándar el resultado. Cada caso de prueba consistirá en dos líneas, la primera con el número de elementos del vector (hasta 300.000) y la segunda con los elementos del vector en sí. Para cada caso de prueba el programa imprimirá un único número con la respuesta. La entrada finalizará con una línea con `-1`. A continuación se muestra un ejemplo de entrada/salida:

Entrada	**Salida**
6 5 6 5 3 2 3 4 10 12 8 2 4 30 30 30 30 -1	16 12 30

2. (3 puntos) La empresa RapidLight se dedica al transporte de mercancias por carretera. La empresa dispone de vehículos con distintas capacidades de depósito, por lo que está interesada en disponer de un programa que, dado un plan de transporte, les permita determinar la capacidad mínima del vehículo necesaria para llevarlo a cabo. Más concretamente, un plan de transporte consta de:

- Una serie de etapas que deben completarse (hasta 100.000). Cada etapa requiere, para su realización, un determinado volumen de combustible. Asimismo, al final de cada etapa será posible, si fuera necesario, volver a llenar el depósito del vehículo.
- Un número máximo de repostajes permitido, ya que cada repostaje supone invertir tiempo, y uno de los objetivos de RapidLight es transportar las mercancias en el menor tiempo posible.

Se pide desarrollar un algoritmo de *divide y vencerás* que determine la capacidad mínima del vehículo necesaria para ejecutar un plan de transporte, dados: (i) la lista de consumos requeridos por las etapas que componen el viaje; y (ii) el número máximo de repostajes permitidos en el camino. Aparte de desarrollar el algoritmo, deberás determinar razonadamente su complejidad. Ten en cuenta que el camión comenzará su andadura con el depósito lleno y le basta con llegar con la gasolina justa a su destino.

El algoritmo se utilizará desde un programa de prueba que lea, por la entrada estándar, planes de transporte, y, para cada plan leído, invoque al algoritmo, y escriba la capacidad mínima necesaria del vehículo para llevar a cabo dicho plan. Cada plan de viaje consta de dos líneas: En

la primera aparece el número de etapas del viaje y el número máximo de repostajes permitido. En la segunda aparecen, en orden, el consumo de combustible requerido por cada etapa. La entrada termina con una línea que contiene -1.

Entrada	**Salida**
2 1 6 4 3 0 3 6 5 3 1 3 6 5 -1	6 14 9

3. (4 puntos) En la villa HappyParties el clan de Los Soprinos celebran su fiesta anual. Tras la finalización de la fiesta, toca volver a casa. Para ello, el clan dispone de una serie de vehículos. Es necesario decidir quién va a ir en cada vehículo. Para ello, deben cumplirse las siguientes condiciones:

1. No puede sobrepasarse el aforo de cada vehículo
2. En cada vehículo debe ir, al menos, una persona que no haya ingerido alcohol durante la celebración (con el fin de que haya alguien que pueda conducir).
3. Para evitar problemas dentro de los coches, el número de consumidores de alcohol en cada uno no puede superar la mitad de la capacidad del vehículo (por ejemplo, si la capacidad de un coche es cuatro o cinco, como mucho podrá haber dos personas que hayan consumido alcohol).
4. No puede dejarse ningún vehículo en la villa (deben haberse desplazado todos).
5. Obviamente, tampoco puede dejarse a ningún miembro del clan sin montar en ningún vehículo.

Cualquier asignación de miembros del clan a vehículos que cumpla dichas condiciones se denomina una asignación *factible*.

Nos han pedido desarrollar un programa que, dados distintos supuestos, utilice un algoritmo de vuelta atrás para contar *todas* las asignaciones factibles de personas a vehículos. Dicho programa leerá una serie de supuestos por la entrada estándar, y, para cada uno de ellos, imprimirá por la salida estándar el resultado. Cada supuesto consistirá en:

- Una primera línea indicando el número de vehículos y el número de personas.
- Una segunda línea con la descripción de cada vehículo: un entero que indica el número máximo de personas que caben en él.
- Una tercera línea con la descripción de cada persona: un valor que puede ser `0`, o `1`, (`1`) indicando que ha consumido alcohol, (`0`) indicando que no ha consumido alcohol.

La entrada finalizará con una línea que contenga -1.

A continuación se muestra un ejemplo de entrada / salida:

Entrada	**Salida**
3 3 1 1 1 0 0 0 2 4 3 3 0 1 0 1 -1	6 4